Exact and Local Search Algorithms to Minimize Multicriteria Scheduling Problem

Anmar S. Al-Tameemi [a] , Dr. Adawyia A. Al-Nuaimi [b]
[a] Minstry of Education, Diyala, Iraq (anmarsapri@gmail.com)
[b] College of Science-Department of Mathematics, Diylala,Iraq (Dr.adawiya@sciences.uodiyala.edu.iq)

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract: In this article , we consider the Multicriteria scheduling problem on a single machine for minimizing the sum of total completion time ($\sum C_j$) with the total tardiness ($\sum T_j$) and maximum earliness (E_{max}). We propose a branch and bound (BAB) algorithm to find the optimal solution for the problem. In this BAB algorithm, a lower bound (LB) based on the decomposition property of the Multicriteria problem is used. Two local search algorithms, descent method (DM) and simulated annealing (SA) are applied for the problem. The algorithms DM and SA are compared with the BAB algorithm in order to evaluate effectiveness of the solution methods. Conclusions are formulated on the efficiency of the algorithms, Based on findings of computational experiments.

Keywords: Multicriteria, single machine, Optimal solution, Scheduling, Local search algorithms.

1. Introduction

Generally, the sequencing problem is denoted as a problem of assigning a set of jobs to a set of machines in time under given constraints [6,7,8]. Jobs j (j=1,2,…,n) are mainly distinguished by processing times (P_j), due dates (d_j), define completion times ($c_j = \sum_{i=1}^{j} p_i$) for particular schedule of jobs. The quality of a schedule can be evaluated by different performance measures including due date's from, the informal Criteria that are applied by practitioners]1[. Real world problems happen in various application domains are usually strictly related to time]1[. In simultaneous Multicriteria problems, when the criteria are weighted differently, an objective function can be defined as the sum of weighted functions and transform the problem into a single criterion sequencing problem. Al-Nuaimi]2[proposed an algorithm to find efficient solutions for Multicriteria scheduling problem of total completion time ($\sum C_j$) with maximum late work (V_{max}) and maximum lateness (L_{max}) on a single machine. In]3 [Al-Nuaimi presented some algorithms to find exact and best possible solutions for the problem of three objectives maximum lateness (L_{max}), maximum earliness (E_{max}) and sum of completion time ($\sum C_j$) in hierarchical case. Also, Al-Nuaimi]4[proposed an algorithm to solve the problem $1//F(\sum C_j , \sum T_j , L_{max})$ to find the set of efficient solutions.

2. Formulation of the problem:

A set N={1,2,…,n} of n independent jobs are available for operating at time zero, each job j (j=1,2,…,n) is to be processed without interruption on single machine that can be handle only one job at a time, requires processing time P_j and due date d_j. For a given sequence δ *of the jobs,* completion time $C_{\delta(j)} = \sum_{i=1}^{j} p_{\delta(i)}$, total tardiness $\sum_{j=1}^{n} T_{\delta(j)}$ where $T_{\delta(j)} = \max\{C_{\delta(j)} - d_{\delta(j)} ,0\}$ and maximum earliness $E_{max}(\delta) = \max\{E_{\delta(1)}, E_{\delta(2)}, \dots, E_{\delta(n)}\}$, can be computed where $E_{\delta(j)} = \max\{d_{\delta(j)} - C_{\delta(j)} ,0\}$. The objective is to schedule the jobs so that the objective function $\sum C_j + \sum T_j + E_{max}$ of three criteria is minimized. This problem is NP-hard since the $\sum_{j=1}^{n} T_j$ is NP-hard. This problem is symbolled by (P) and can be formulated as followed:

$$Z=\{ \sum_{j=1}^{n} C_{\delta(j)} + \sum_{j=1}^{n} T_{\delta(j)} + E_{max}(\delta)$$

s.t.

$$C_{\delta(1)} = P_{\delta(1)}$$

$$C_{\delta(j+1)} = C_{\delta(j)} + P_{\delta(j+1)} \qquad j=1,2,\dots,n-1 \qquad \dots\dots(P)$$

$$T_{\delta(j)} \geq C_{\delta(j)} - d_{\delta(j)} \qquad j=1,2,\dots,n$$

$$T_{\delta(j)} \geq 0$$

$$E_{\delta(j)} \geq d_{\delta(j)} - C_{\delta(j)} \qquad j=1,2,\dots,n$$

$$E_{\delta(j)} \geq 0$$

Where S is the set of all schedules.

3. The Branch and Bound (BAB) algorithm [5].

The Branch and Bound (BAB) algorithm is an enumeration method for finding the best solution by systematically evaluating subsets of possible solutions. All s ∈ S are implicitly enumerated by examining smaller subsets of the set of feasible solutions S. S* is discovered by the branch and bound. These subsets can be thought of as collections of solutions to the original problem's corresponding sub problems.

The steps of the BAB algorithm are as follows:

1. The branching step:

This procedure explains how to divide possible solutions into subsets. These subsets can be thought of as a collection of solutions to the original problem's corresponding sub problems.

2. The bounding step:

This procedure explains how to calculate a lower bound (LB) on the value of the optimal solution for each sub problem that is created during the branching process.

3. The search strategy step:

This refers to the method for branching from a node in the search tree. Among the nodes that have recently been formed. Normal

The BAB algorithm will now be given a formal definition. The total number of possible sequences is divided into disjoint subsets, each of which can contain multiple sequences. We measure a (LB) for each subset, which is the cost of the sequencing jobs (tasks) (depending on the objective function) and the cost of the unsequencing jobs (depending on the objective function) (depending on the derived LB). When the upper bound's (LB) is greater than or equal to the subset's (LB) (UB). This subset is ignored (the value for a trial solution is the upper bound UB) (schedule). Because any subset with a value less than UB can only occur in the remaining subsets, the trial solution is obtained using a heuristic method. These remaining subsets must be examined one by one.

In accordance with a quest plan. One of these subsets is chosen. This subset is then partitioned into smaller disjoint subsets (as seen above). A full schedule of the jobs should exist as soon as one of these subsets contains only one element. If the value of this schedule is less than the current upper bound UB, the UB is reset to take that value. After that, the process is repeated until all (node) subsets have been taken into account. The optimal solution for the problem is the upper bound (UB) at the end of this BAB process.

The BAB algorithm uses the upper bound of the objective value (UB) to truncate search tree branches that do not lead to an optimal solution. UB is equal to the objective value for the best solution constructed by BAB during the search. The initial upper bound is determined by the problem at hand at the start of the solution process.

At the root node of the BAB search tree, the heuristic method which is applied once to find the upper bound (UB) on the minimum value of ($\sum_{j=1}^{n} Cj + \sum_{j=1}^{n} Tj + E_{max}$) is obtained by the shortest processing time (SPT) rule, that is sequencing the jobs in non-decreasing order of their processing time (P_j), j=1,2,…,n.

To calculate a lower bound (LB) for each node, let δ be the sequencing jobs and δ' be the unsequencing jobs, hence

LB(δ)=Exact cost of (δ)+ cost of (δ').

Where cost of δ' is obtained by using lower bounding procedure.

Decomposing the problem into three sub problems (SP_1),(SP_2) and (SP_3) as follows:

$Z_1 = \{\sum_{j=1}^{n} C_{\delta(j)}\}$

s.t. (SP1)

$C_{\delta(1)} = P_{\delta(1)}$

$C_{\delta(j)} = C_{\delta(j-1)} + P_{\delta(j)}$, j=2,3,...,n

This sub problem (SP_1) is solved by (SPT) rule.

$Z_2 = \{\sum_{j=1}^{n} T_{\delta(j)}\}$

s.t. (SP_2)

$T_{\delta(j)} \geq C_{\delta(j)} - d_{\delta(j)}$

$T_{\delta(j)} \geq 0$

This sub problem (SP_2) is NP-hard.

$Z_3 = \min_{\delta \in S} \{E_{max}(\delta)\}$

S.t. (SP_3)

$E_{\delta(j)} \geq d_{\delta(j)} - C_{\delta(j)}$, j=1,2,...,n

$E_{\delta(j)} \geq 0$

This sub problem (SP_3) is solved by minimum slack time (MST) rule, that is sequencing the jobs in non-decreasing order of their slack time $d_j - P_j$,J=1,2,…,n.

Thus, the lower bound (LB) for the problem (P) is the sum of minimum values of the sub problems (SP_1), (SP_2) and (SP_3). We proposed that the minimum value for $\sum T_j$ is obtained by $\sum T_j$ (SPT) – T_{max} (EDD), Where EDD is the earliest due date value, i.e., sequencing the jobs in non-decreasing order of their due dates.

It is clear that $\sum T_j$ (SPT) – T_{max} (EDD) $\leq \sum T_j$ Let Z_1, Z_2 and Z_3 be the minimum values of (SP_1), (SP_2) and (SP_3), then applying the following theorem to get a lower bound for (P).

Theorem (3.1) [5].

If Z_1, Z_2, Z_3 and Z are the minimum objective function values of (SP_1), (SP_2) and (SP_3) and (P) respectively then $Z_1+Z_2+Z_3 \leq Z$.

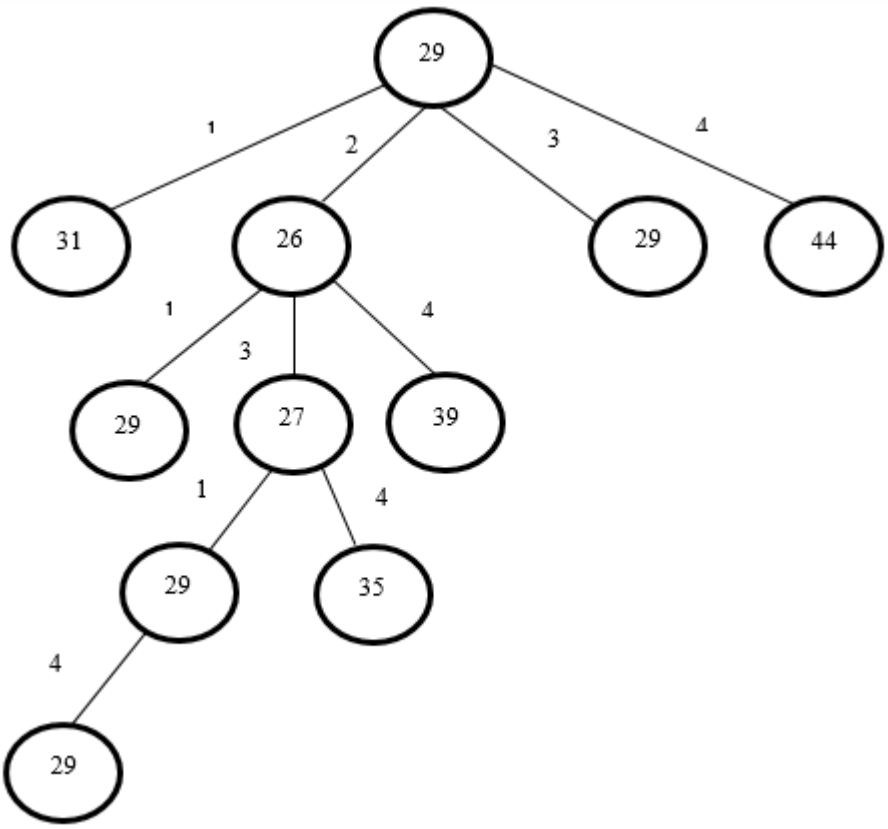
By using theorem (3.1) a lower bound (LB) for the problem (P) is given by

$LB = Z_1+Z_2+Z_3$.■

An example (1): Suppose the problem (P) has the following data:

j	1	2	3	4
P_j	2	3	1	6
d_i	8	4	6	10

The BAB algorithm tree to find the optimal solution for the problem (P) is as follows:



The optimal sequence is (2,3,1,4) with the optimal value 29.

4. The local search algorithms.

The local search algorithms can find the best near optimal solution within a reasonable running time. Approximation local search is a collection of methods that iteratively search through the set of solutions. Beginning from an initial solution, which is found by heuristic method, a local search procedure moves from one feasible solution to a neighbour solution until some termination criteria are met. The choice of suitable neighbourhood function has an important influence on the performance of a local search. These neighbourhood functions define the set of solutions to which the local search procedure may move to a single iteration [9]. This is formulated in the following definition:

Definition 1 [1]:

A pair (S,f) is an instance of a combinatorial optimization problem , where the cost function F:S $\rightarrow$ R and the solution set S is the set of all feasible solutions.

The problem is to find a global (minimal) optimal solution. i.e. an $s^* \in S$, such that $F(s^*) \leq F(s)$ for all $s \in S$.

4.1 The Solution representation [1].

The representation of solution depends on the problem specification. In a sequencing problem of n tasks (jobs). A solution is denoted by a Permutation of the integers 1,2,…,n.

Definition 11[1]: A mapping N^*:S $\rightarrow$ P(S) which specifies for every b $s \in S$ a subset $N^*(S)$ of S neighbours of s is called a neighbourhood function N^*

For scheduling problems, the representation is a permutation of the integers 1,2,…,n where n is the number of jobs. Two basic neighbourhoods can be defined [9]. With this representation. Each of which is determined by considering a typical neighbour of the schedule (1, 2, 3, 4, 5, 6, 7, 8).

1- Shift (insert): Pick up a job from position I in the schedule and insert it at position j (either before (left insert) after (right insert) the original position). Thus (1,6,2,3,4,5,7,8) and (1,2,3,4,5,7,6,8) are both neighbors.

2-Interchange: Swap two jobs (i,j) which may not be adjacent. Thus (1, 6, 3, 4, 5, 2, 7, 8) is a neighbour.

Definition 111 [1]: Let (S, F) be an instance of a combinational minimization problem and let N^* be a neighbourhood function. The neighbourhood function N^* is called exact if every local minimum with respect to N^* is called exact if every local minimum with respect to N^* is also a global minimum. A solution $s^* \in S$ is called a local minimal solution with respect to N* if $F(s^*) \leq F(s)$ for all $s \in N^*(s^*)$.

4.2 Descent method (DM) [1].

Only moves that result in an improvement in the value of objective function value are accepted. This method is the simplest kind of neighbourhood search. Which is sometimes known as iterative local improvement. In this method.

A descent method has the following main component:

1. **Initialization**:
In this step, the method has to be started with an initial solution. This solution can be composed by some heuristic method or it can be chosen at random.

2. **Neighborhood Generating :**
To generate a neighborhood, the two basic neighborhoods swap (insert) which are explained in section 3.1 can be used.

3. **Criterion Termination:**
There are many methods for termination criterion of the algorithm. In this paper, the one that is used in a constant number this of iterations; i.e, the algorithm is stopped after 18000 iteration with approximate solution.

4.3 Simulated annealing (SA) algorithm [1].

This algorithm is known as the probabilistic approach, and it is used to solve problems involving combinatorial minimization. This algorithm employs a probabilistic acceptance law. Acceptance is given to those that result in an improvement in the value objective function or leave the value unchanged. That is, a step that increases the value of the objective function by Δ is agreed with a probability of $\exp(-\Delta/T)$ where T is the temperature a parameter. The value of T fluctuates during the quest. T starts with a high value and progressively decreases.

A simulated annealing has the following main components:

1. **Initialization.**
The starting solution can be generated using a heuristic algorithm or chosen at random in this stage. This will be the first current solution for the (SA) algorithm, with Z serving as the value objective function.

2. **Generation of neighborhood.**

Two basic neighbourhoods swap or insert which are explained in section 3.1 can be used to generate a neighbourhood.

3. **Test of accepting**

The difference value between the new value Ź and the initial current solution Z, $\Delta = Ź - Z$ is computed and evaluated in the following steps:

i) Z` is accepted as new current solution with setting Z=Ź If $\Delta \leq 0$.

ii) Z` is accepted with $p(\Delta) = exp(-\Delta/T)$, which is the probability of accepting a move If $\Delta > 0$.

4. Stopping criterion

The method is stopped with approximate solution when the iteration reaches 1800.

4. Computational results.

The BAB algorithm and local search algorithms are run on a personal computer after being coded in Matlab R2014b. Test problems are created as follows: as a result of using the discrete uniform distribution [1,10], an integer processing time Pj is provided for each job j. For each job j. P (1-TF+RDD/2), a discrete uniform distribution [P (1-TF-RDD/2) is also used to generate an integer due date, where P is a function of the

average tardiness factor (TF) and the relative range of due dates (RDD).For two parameters, the specific values 0.2,0.4,0.6,0.8,1.0 are taken into account. Two problems are produced for each of the five values of the parameters producing for each of the selected values of n. Where the number of jobs n = 5,7,9,11,13. The following tables give the comparative of computational results and the time (in seconds) for the problem (P). A problem is abandoned if it cannot be solved to its optimality within 1800 seconds. We have in every single one of these tables.:

Ex: Number of example.

Node: Number of nodes.

Optimal: The optimal value that is obtained by BAB algorithm.

No.of opt.: Number of examples that catches the optimal value.

No.of best: Number of examples that catches the best value.

DM: The value that is obtained by decent method.

SA: The value that is obtained by simulated annealing method.

Time: Time in seconds.

Status= ⌈ **1, if the example is solved**
1 ⌊ **, otherwise**

Table (1): A comparison between the optimal solutions obtained by BAB algorithm and the values result of local search algorithms at n=5.

BAB						Local Search			
n	Ex	Optimal	Node	Time	Status	DM	Time	SA	Time
5	1	74	98	0.058544	1	74	0.434974	74	0.442241
	2	101	52	0.003252	1	101	0.437393	101	0.441452
	3	134	307	0.009876	1	134	0.42905	134	0.433851
	4	74	73	0.002755	1	74	0.432543	74	0.438852
	5	147	195	0.006027	1	147	0.432276	147	0.444559
	6	94	156	0.004961	1	94	0.438341	94	0.448514
	7	144	196	0.006223	1	144	0.43245	144	0.435306
	8	126	110	0.003537	1	126	0.430599	126	0.432089
	9	128	132	0.004153	1	128	0.434114	128	0.433752
	10	128	140	0.004861	1	128	0.432461	128	0.437437
		No. of optimal			.	10		10	

Table (2): A comparison between the optimal solutions obtained by BAB algorithm and the values result of local search algorithms at n=7.

BAB						Local Search			
n	Ex	Optimal	node	Time	Status	DM	Time	SA	Time
7	1	149	576	0.074127	1	149	0.459881	149	0.464976
	2	181	1649	0.049056	1	181	0.45969	181	0.464232
	3	155	1865	0.054323	1	155	0.46471	155	0.45741
	4	319	7296	0.215803	1	331	0.473887	319	0.460469
	5	159	993	0.030626	1	159	0.463881	159	0.473351
	6	181	1444	0.044574	1	181	0.446838	181	0.453018
	7	163	1013	0.029366	1	163	0.450523	163	0.456051
	8	185	1401	0.044981	1	185	0.45284	185	0.451961
	9	108	5168	0.153238	1	108	0.4562	108	0.454042
	10	227	1853	0.053289	1	227	0.451608	227	0.462075
		No. of optimal			.	9		10	

Table (3): A comparison between the optimal solutions obtained by BAB algorithm and the values result of local search algorithms at n=9.

BAB						Local Search			
n	Ex	Optimal	node	Time	Status	DM	Time	SA	Time
9	1	298	1512	0.103231	1	298	0.53425	298	0.536331
	2	206	19404	0.669483	1	206	0.57251	206	0.536413
	3	200	46645	1.29119	1	200	0.591698	200	0.562955
	4	341	47239	1.304335	1	341	0.551664	341	0.565978
	5	196	16954	0.461582	1	196	0.541463	196	0.570218
	6	304	24716	0.726086	1	304	0.545333	304	0.53622
	7	140	16888	0.500476	1	140	0.538294	140	0.556678
	8	378	98906	2.8047	1	378	0.602471	378	0.535638
	9	322	27287	0.743506	1	322	0.519206	322	0.526612
	10	301	30011	0.829983	1	301	0.535423	301	0.568356

		No. of optimal			.	10		10	

Table (4): A comparison between the optimal solutions obtained by BAB algorithm and the values result of local search algorithms at n=11.

BAB						Local Search			
n	Ex	0ptimal	Node	Time	Status	DM	Time	SA	Time
11	1	380	862639	25.53819	1	282	0.561815	380	0.556558
	2	614	4883141	146.780577	1	627	0.568649	616	0.608228
	3	436	1138123	34.5071715	1	437	0.563739	436	0.557962
	4	340	225082	7.0207728	1	342	0.560996	340	0.565659
	5	393	666300	19.4370768	1	393	0.558404	393	0.547713
	6	528	284273	8.5266774	1	528	0.558589	528	0.561838
	7	451	722008	22.042973	1	452	0.582013	453	0.646561
	8	284	483145	13.8482672	1	284	0.559388	284	0.555403
	9	478	846894	25.8682167	1	478	0.54741	478	0.570294
	10	588	3435594	103.426301	1	588	0.550936	588	0.549352
		No. of optimal			.	5		8	

Table (5): A comparison between the optimal solutions obtained by BAB algorithm and the values result of local search algorithms at n=13.

BAB						Local Search			
n	Ex	0ptimal	Node	Time	Status	DM	Time	SA	Time
13	1	792	43497668	1329.71367	1	797	0.718032	795	0.70948
	2	569	25103067	774.251406	1	570	0.582725	570	0.606381
	3	483	24079250	733.735107	1	483	0.580494	485	0.578516
	4	490	9327944	342.161331	1	490	0.603786	490	0.588358
	5	645	41447845	1800.00002	0	645	0.708157	648	0.58716
	6	689	40341289	1800.00003	0	689	0.641504	691	0.70488
	7	725	40336942	1800.00011	0	725	0.578152	730	0.604362
	8	586	20875539	940.108954	1	586	0.674821	588	0.609936
	9	485	20710738	876.02978	1	485	0.580647	486	0.576686
	10	846	41276248	1800.00009	0	846	0.593704	846	0.576091
		No. of optimal			.	8		2	

Table (6): The values result of local search algorithms at n**=100.**

Ex	Best	DM	Time	SA	Time
1	28260	28260	1.688922	28778	2.146758
2	32288	32288	1.841758	32983	1.731629
3	33142	33142	1.74576	33791	1.741633
4	25434	25434	1.780215	26146	1.830915
5	26484	26484	1.704717	27173	1.73558
6	33294	33294	1.6558	33380	1.696299
7	33484	33484	1.653292	33610	1.786976

8	32348	32348	1.675779	32797	1.73325
9	29396	29396	1.917032	29738	1.721954
10	28397	28397	1.691821	28681	1.730499
	No. of best	10		0	

Table (7): The values result of local search algorithms at **n=500.**

Ex	Best	DM	Time	SA	Time
1	713688	713688	6.643484	721116	6.60786
2	660697	660697	7.386148	673628	6.712611
3	752493	752493	6.660413	757187	6.864889
4	738057	738057	6.739151	746586	6.774209
5	675798	675798	6.672325	684939	6.650751
6	735238	735238	6.830684	740941	6.63883
7	750107	750107	6.729234	754600	6.993623
8	759316	759316	7.727227	764393	6.946161
9	889461	889461	8.181189	890449	7.294928
10	932939	932939	6.915933	932948	7.225786
	No.of best	10		0	

Table (8): The values result of local search algorithms at **n=1000.**

Ex	Best	DM	Time	SA	Time
1	2571573	2571573	13.57011	2578112	13.10906
2	2951210	2951210	13.54264	2953360	13.71777
3	2996746	2996746	13.81543	2999152	13.05414
4	2842043	2842043	12.93417	2844309	12.84052
5	2906018	2906018	13.30481	2910830	12.87375
6	3174167	3174167	12.7083	3175709	12.73979
7	3485121	3485121	12.7705	3486448	12.7749
8	2895839	2895839	13.32522	2900250	13.00193
9	3373825	3373825	12.88475	3376331	12.88404
10	3217738	3217738	13.27027	3219015	13.02563
	No. of best	10		0	

Table (9): The values result of local search algorithms at **n=5000.**

Ex	Best	DM	Time	SA	Time
1	73536615	73536615	60.83256	73536865	60.87721
2	63254813	63254813	61.5373	63255660	61.92102
3	67113499	67113499	61.44035	67113824	60.814
4	79686070	79686070	60.95324	79686221	59.77198
5	68036362	68036362	61.95672	68036831	60.51956
6	70491796	70491796	60.8792	70492253	61.22687
7	73448798	73448798	60.36676	73449028	60.53428
8	91175799	91175799	61.24575	91175828	58.74872
9	93941994	93941994	59.54845	93941995	59.82845
10	76722028	76722028	60.44667	76722261	60.69898
	No. of best	10		0	

Table (10): The values result of local search algorithms at n**=10000.**

Ex	Best	DM	Time	SA	Time
1	289606034	289606034	155.2902	289606135	147.3921
2	273653864	273653864	153.0772	273653987	145.3847

3	285225751	285225751	154.83	285225807	147.4262
4	284408782	284408782	162.3384	284408938	151.158
5	276890195	276890195	154.9541	276890298	162.6316
6	336633779	336633779	137.8737	336633828	144.5711
7	321570865	321570865	143.1081	321570917	135.8353
8	346921027	346921027	146.2184	346921063	145.0729
9	299196329	299196329	145.0793	299196441	147.0557
10	305059249	305059249	147.2527	305059347	146.8354
	No.of best	10		0	

Table (11): The values result of local search algorithms at **n=20000.**

Ex	Best	DM	Time	SA	Time
1	1198810298	1198810298	237.7489	1198810352	240.5755
2	1166318774	1166318774	240.2549	1166318816	235.574
3	1035314962	1035314962	242.8797	1035315030	239.9499
4	1166864162	1166864162	238.9088	1166864226	240.2091
5	1234855812	1234855812	239.3137	1234855850	236.8713
6	1170262797	1170262797	239.1409	1170262828	237.8809
7	1146405539	1146405539	240.5716	1146405586	241.4909
8	1246434331	1246434331	237.116	1246434352	235.1815
9	1366225692	1366225692	235.0879	1366225709	233.0021
10	1545672927	1545672927	229.7283	154672927	228.244
	No.of best	10		1	

Table (12): The values result of local search algorithms at n=**300000.**

Ex	Best	DM	Time	SA	Time
1	2316579635	2316579635	368.9691	2316579694	365.4173
2	2371616071	2371616071	7000.079	2371616117	367.3791
3	2754714509	2754714509	493.5022	2754714539	492.3983
4	2912393595	2912393595	477.308	2912393611	472.8284
5	2568730859	2568741545	10916.23	2568730859	1637.515
6	3082704178	3082704178	386.4808	3082704201	600.0024
7	3073066706	3073066706	372.3755	3073066715	371.7001
8	2838453378	2838453378	363.0306	2838453387	356.1615
9	2906051952	2906051952	470.5902	2906051976	356.743
10	3165132308	3165132308	481.673	3165132313	468.6973
	No. of best	9		1	

Table (13): The values result of local search algorithms at **n=400000.**

Ex	Best	DM	Time	SA	Time
1	4517342015	4517342015	468.3425	4517342036	463.6928
2	4342910570	4342910570	483.8663	4342910604	468.9732
3	4697900554	4697900554	463.0457	4697900560	466.0637
4	4239059324	4239059324	480.22	4239059359	467.2866
5	5175147897	5175147897	466.3659	5175147903	461.4844
6	4470139888	4470139888	505.815	4470139898	465.3543
7	4819997499	4819997499	463.9291	4819997510	466.0675
8	4717213474	4717213474	477.5863	4717213490	461.3611
9	5441662736	5441662736	460.291	5441662746	459.9377
10	5058154973	5058154973	463.9767	5048154987	460.7368
	No. of best	10		0	

6. Conclusions:

This paper proposes an effective branch and bound (BAB) algorithm to find the best solution to the problem of reducing a $\sum_{j=1}^{n} Cj + \sum_{j=1}^{n} Tj + E_{max}$ .On a large number of test problems, the (BAB) method is used. The BAB algorithm is efficient, as evidenced by the computed values. Finding approximation solutions for the problem can also be achieved by applying both simulated annealing (SA) and local search algorithms descent method (DM). On a broad set of test problems, a computational experiment for local search algorithms is presented. The descent method (DM) is much more successful for problems of large size n=5000,10000,20000,30000,40000. This is the most important we can derive from our computational results.

An interesting future research topic would include the development of the lower bound (LB) and experimentation with Meta heuristic algorithms.

References (APA)

[1] Al- Nuaimi A. A. M. , Local search algorithms for multiobjective scheduling problem, Journal of AI-Rafidain University College 36: 201-217,2015.

[2] Al- Nuaimi A. A. M. , A proposed algorithm to find efficient solutions for Multicriteria problem, Journal of Engineering and Applied Sciences 14(2): 5547-5549, 2019.

[3] Al- Nuaimi A. A. M. , Minimizing three hierarchically Criteria on a single machine, Diyala Journal for pure sciences 13(1): 14-22, 2017

[4] Al- Nuaimi A. A. M. , An algorithm for solving three criteria scheduling roblem on a single machine, Int. J. Agricult. Stat. Sci, 14(1): 271-273,2018.

[5] Al- Nuaimi A. A. M. , Optimal solution for simultaneous multicriteria problem, Diyala Journal for pure Sciences 12(2): 18-27,2016.

[6] Blazewicz J., Ecker K., Pesch E., Schmidt G., Weglarz J., Hand book on scheduling. From theory to applications, Berlin-Heidelberg- New York, Springer, 2007.

[7] Bruker P., Scheduling algorithms, Berlin-Heidelberg- New York, Springer, 2007.

[8] Pinedo M., Scheduling: theory, algorithms and systems, New York, Springer, 2008.